CPRE 288
Reading Preparation for Using the TM4C GPTM in Lab
GPTM: General Purpose Timer Module
Input Edge-Time Mode (also called Input Capture)

Usually preparation for the next lab begins in class meetings during the week before the lab. This is intended to help students have some foundation for the prelab and lab programming in advance. The concepts are then delved into more deeply during the week of the lab, with the intent for students to make connections, get practice, ask questions, etc. – essentially get momentum while proceeding into and through lab that week.

The following preparation will be very helpful before doing Prelab 9 to so that you know more, or know where to find more, about input capture and the TM4C GPTM Timer module, in addition to the Tiva datasheet.

Refer to basic timer functionality as introduced in class and given in notes, slides and readings:

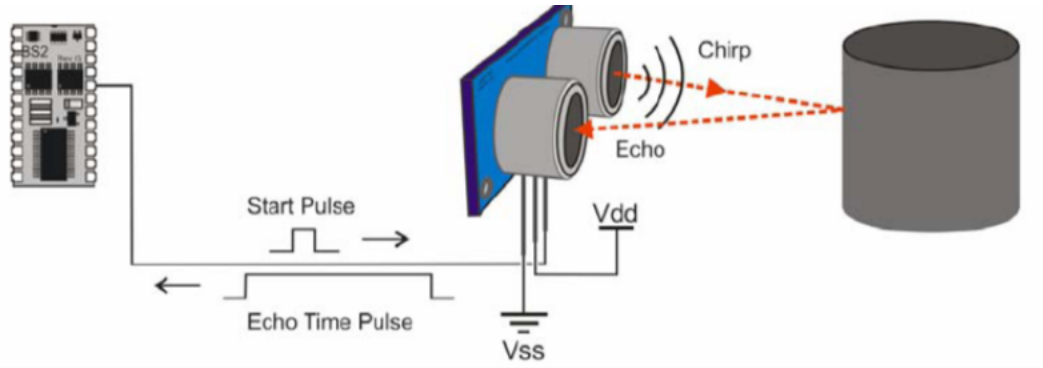
1. mtg-notes-concepts-timers-GPTimerIC.pdf
2. Readings – Timers page in Canvas
3. Introduction-Timers-Silva.pdf

Prep Time: about 30-40 minutes (need not be done in one sitting)

There is not a VYES book chapter on the GPTM. VYES uses a different system timer. Thus we will be using the Bai book and Tiva datasheet as our primary resources.

In lab, you will use an ultrasonic sensor to measure distance to an object. The sensor provides a pulse signal to the microcontroller (echo time pulse) that represents the time for a sound wave to travel from the sensor to an object and be reflected back. Thus the microcontroller needs to measure the width of the pulse, or the time between the rising edge and the falling edge of the signal from the sensor. This is done using one of the GPTM timers on the microcontroller. The timer uses a special mode called input edge-time mode, also referred to as input capture in other systems.

Below is a diagram of the sensor operation:

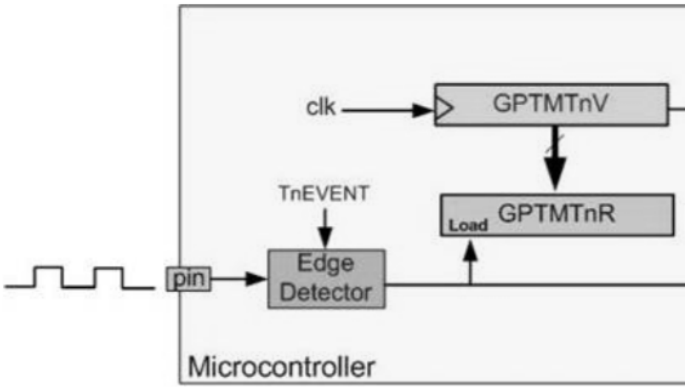


We need to measure the width of the Echo Time Pulse. The first event is its rising edge, and the second event is its falling edge. We will detect the first event, capture its time using the timer, then detect the second event, and capture its time. The time duration is the difference between the captured times of the events. The events occur on an input pin of the microcontroller and are detected using interrupts. As you might expect, the input pin is an alternate function of a GPIO pin. The alternate function is labeled TCCP (Timer Capture, Compare, PWM – we are using the Capture function). The ultrasonic sensor in lab is wired to the Timer 3B input capture pin (T3CCP1).

Ignore the "Start Pulse" in the diagram above for now. You will learn more about that soon enough in lab.

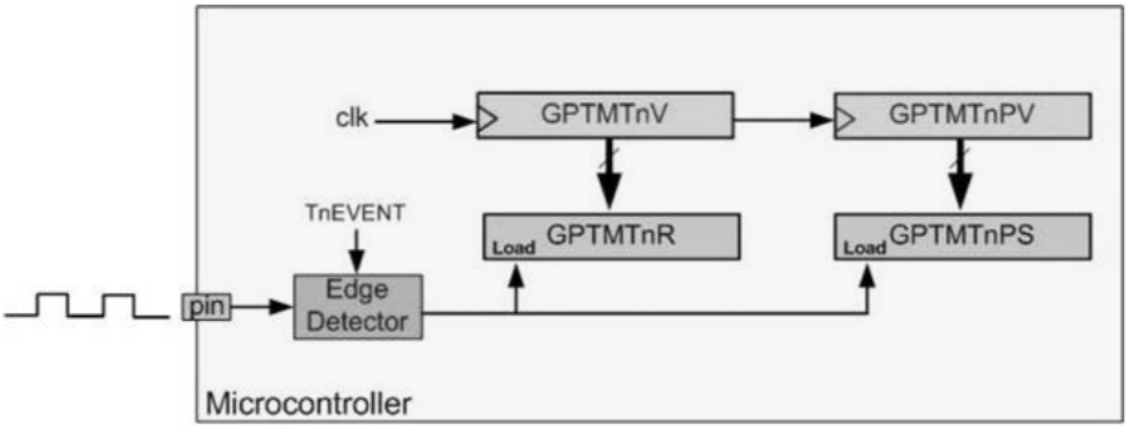
Your job at the moment is to learn about the GPTM timer and how it implements input capture to get the echo pulse width time.

The following figure illustrates the basic principle in which an edge is detected on the input pin, which causes the GPTM timer to load the value of the 16-bit free-running timer (GPTMTV, TV = Timer Value) into the 16-bit GPTMTR register (TR = Timer Register). It's that simple (almost).



We want to extend the 16-bit timer to 24 bits so that we have a longer range for the timer. The figure below shows the extension of the timer register to 24 bits using an 8-bit prescaler (PV or

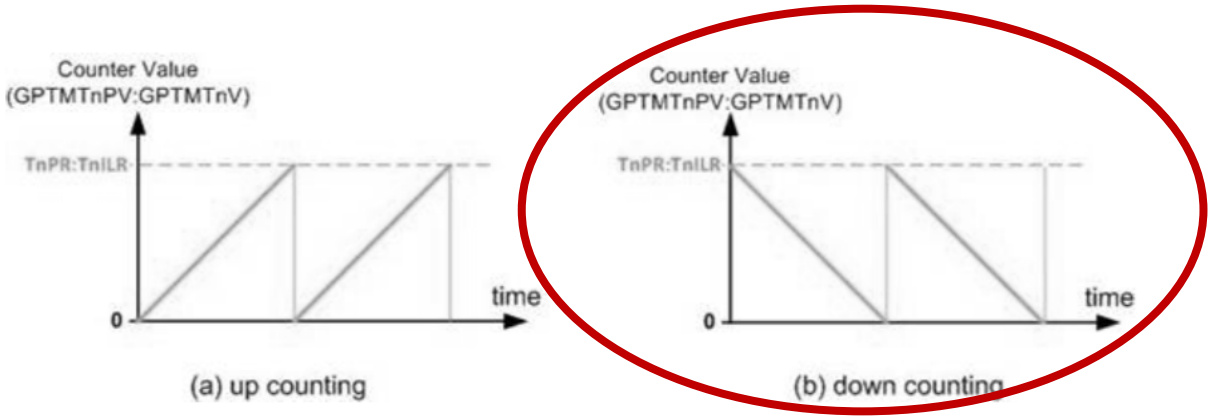
PS) register. Thus the 24-bit timer value consists of 8 bits from the prescaler (most significant byte) and 16 bits from the regular timer, resulting in a 24-bit value:
GPTMTPS[23..16] : GPTMTR[15..0].
For example, if GPTMTPS = 0x13 and GPTMTR = 0xAF50, the result is a 24-bit timer value of 0x13AF50.



(/edwiki/File:Tm4c_input_edge_time_capture.png)

Figure 9.1: Input Edge Time Capturing

Remember that the free-running timer (GPTMTPV:GPTMTV) looks something like this when counting up or counting down. In lab, you will configure the timer to count down from an initial value to 0 (the timeout or overflow value). The initial value is given in another pair of registers (GPTMTPR:GPTMTILR)



(/edwiki/File:Tm4c_counting_in_input_edge_time_mode.png)

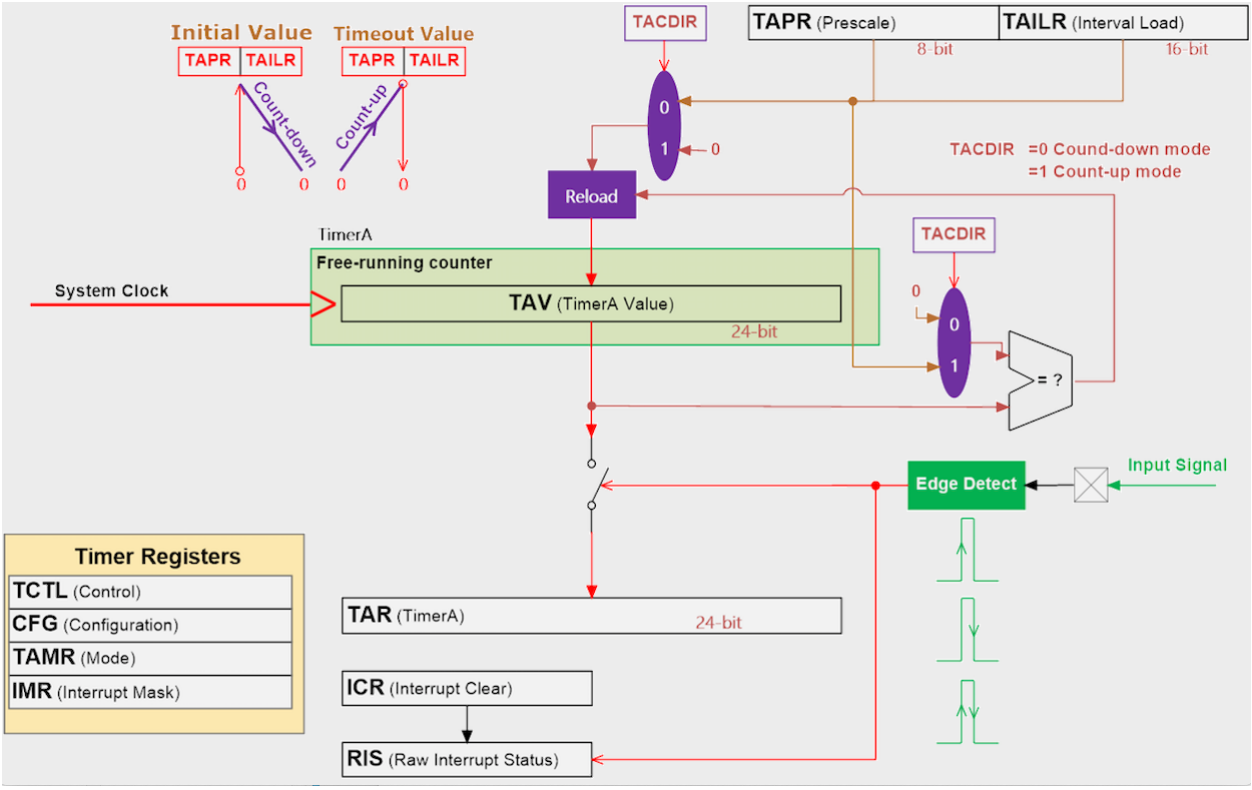
Figure 9.2: Counting in Input Edge-Time Mode

Note: the "n" in the figures above would be replaced with A or B, because each timer module has two timers, Timer A and Timer B. Thus all acronyms need to specify A or B. The ultrasonic sensor in lab is wired to Timer 3B.

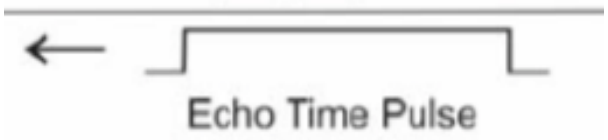
Of course, you as the programmer need to configure the timer (for lab, that is Timer 3B). You will use various registers to initialize the following:

- 16-bit timer (you start with this, and then extend it to 24)
- Capture mode
- Edge-time mode
- Count direction (down)
- Event type (both rising and falling edges)
- Use of prescaler to extend the timer to 24 bits
- Timer initial value (e.g., counts from this value down to 0)
- Interrupts

The figure below depicts the above information with some additional details (shown with Timer A for the register names). Start with the "Input Signal" and trace the connections to registers.



Your job is to understand the GPTM registers shown in the figure above and how they are used to implement input capture to get the sensor's echo pulse width time.



Preparation: Browse selected sections of Bai book, Chapter 9 (highlighted and annotated) (found on the Readings – Timers page in Canvas): Bai-book-Chapter9-TimerIC.pdf
The original chapter is found on the Textbooks page in Canvas.

See section 9.2.6.3 on Initialization and Configuration in the Bai book (similar to section 11.4.4 in the Tiva datasheet). It gives a list of steps, registers and values to use. Try modifying this list for the lab (i.e., count-down timer starting with the max value, detecting both rising and falling edges, using interrupts).

See Figure 9.14 in the Bai book. This is sample code for input capture; pay attention to obvious differences compared with our lab environment. The functionality in lab is different that this example (no LEDs etc.), but the initialization will have similarities. Try modifying the initialization part of this code for lab (using Timer 3B). One way to get started is by replacing the register variables with the appropriate macro names.

Even if you do not directly use the steps and sample code from the Bai book, it's a good learning exercise to think about the information and compare it to your lab work. It lets you see information in different contexts.

Note: Our GPTM module clock is RCGCTIMER, and you will use PRTIMER to wait for the GPTM to be ready after starting the clock.